REMARKS

Overview

The Examiner responded in the prior Office Action as follows: rejected claims 11-29 and 42-57 under 35 U.S.C. § 102(b) as being anticipated by Theimer et al. (U.S. Patent No. 5,611,050). Applicants have traversed the Examiner's rejections without amending the claims, as discussed below, and thus claims 11-29 and 42-57 continue to be pending.

Embodiments of the Present Invention

Embodiments of the present invention are directed to a software facility that exchanges information between sources of context data and consumers of context data, such as for use in modeling a current context or state by generating values for various attributes of the state. In particular, in one embodiment, a characterization module operating in a wearable computer system receives context information from one or more context servers (or "suppliers"), and provides that information to one or more context clients (or "consumers"). This context information represents a particular context (or "state" or "condition") of the wearable, the user of the wearable, the surrounding physical environment and/or the available electronic data environment (or "cyber-environment"). In some embodiments the context is represented (or "modeled") with a variety of attributes (or "variables") each modeling a single aspect of the context.

The Examiner's attention is respectfully drawn to Applicants' Figure 28, which is a flow diagram of an embodiment of a Context Server (CS) routine. Step 2850 determines if a "pull attribute value" request has been received by the CS, along with an ID that uniquely identifies the request. The unique ID in this example embodiment is used to allow the context server to determine if a circular reference exists when determining the requested attribute value. For example, consider the situation in which CS 1 is registered to supply values for attribute 1, and CS 2 is registered to supply values for attribute 2. In addition, imagine that the calculation of the value for attribute 1 depends on the value for attribute 2. Thus, when CS 1 receives a pull

request to supply the value for attribute 1, it requests the value for attribute 2 from the characterization module (CM), which in turn requests the value from CS 2. After receiving the value for attribute 2 from CS 2 via the CM, CS 1 can then calculate the value for attribute 1 and satisfy the initial request by supplying the value to the CM. If, however, CS 2 also depends on the value for attribute 1 in order to calculate the value for attribute 2, a circular reference may exist. In that situation, after CS 1 requests the value of the attribute 2 in order to calculate the value for attribute 1, CS 2 may in turn request the value of attribute 1 from CS 1 (via the CM), thus creating a loop such that neither attribute value can be calculated. In order to detect such circular references, the example routine uses a unique ID passed along with attribute value requests. Alternative equivalent ways of identifying circular references could instead be used, or no such processing could be performed. In the illustrated embodiment, if it is determined that a "pull attribute value" request has been received by the CS, the routine continues to step 2852 to determine if the received request ID matches a temporarily stored ID for the same attribute value. In the example embodiment of Figure 28, if a circular reference exists, then the routine continues to step 2853 to return an error message.

The Theimer Reference

Theimer is directed to detecting, selecting and controlling computer controlled devices, based on the proximity of the device to the user, the current context of the user, the location of other nearby users and devices, and the current state of the devices. When an application or agent needs to discover the location, or other personal information, of a particular user, it can request that information from that user's agent. When an application or agent needs to determine coincidence of people or devices are at or near a particular location, it uses the Location Service 82. Location-specific information about users and devices is registered in the Location Service 82 by the relevant agents. The Location Service also keeps track of which applications and agents have expressed interest in being notified about changes in the set of users or devices currently at specific locations. When changes occur, the relevant set of interested clients is notified via an RPC callback.

The Examiner's attention is respectfully drawn to Theimer's Figure 10, which describes the Location Service. If the Location Service receives a request for callbacks (step 256), the Location Service registers the request (step 258). For example, a user agent may register a request for notification of any new objects moving into a particular office that the user is in. Whenever a new object is registered, the location of the object is updated (step 250) and registered callback requests that are affected by the updated location are performed (step 254).

Analysis

The Examiner has rejected each of the previously pending claims 11-29 and 42-57 under 35 U.S.C. § 102(b) as being unpatentable over Theimer. However, each of the pending claims as rejected includes features and provides functionality not disclosed by Theimer, and thus is patentable over Theimer.

For example, claims 11 and 25 recite, "during generating of the requested value of the first state attribute by the client-source, monitoring requests from the client-source for values of one or more indicated state attributes needed for the generating of the requested value of the first state attribute . . . and when it is determined that a state attribute indicated in one of the monitored requests is the first state attribute, indicating a presence of a circular reference during the generating of the requested value of the first state attribute." Claim 27 recites similar language.

The Examiner has cited column 10, lines 56-63, and column 12, lines 5-60, of Theimer in arguing that Theimer teaches these claimed elements. The cited excerpts are reproduced below for the Examiner's convenience:

> The step in box 122 exports the RPC interface for UserAgent 100 so that would-be clients can find this UserAgent process and interact with it. This involves, among other things, registering an RPC address under the UserAgent's name with the Name Service. The step in box 124 registers the UserAgent with the location service and registers callbacks with any services which monitor state changes in which the user is interested. (Theimer, 10:56-63.)
>
> ---

Services may also send callback RPCs to the UserAgent when information changes take place with respect to subjects that have been registered as being of interest to the UserAgent. For callback RPCs from other services, in the step in box 134, the UserAgent updates its internal state variables in the step in box to reflect the information it has received in a callback. Moreover, the UserAgent may also execute various other actions if it is required to do so by the combination of the UserAgent's new state and the policy settings specified by the user. Some possible examples of callback RPCs and the actions they might elicit in the UserAgent are:

a) Callback from Badge Service indicating that user was sighted at location X. The UserAgent will update its internal representation of where the user is currently located. If the user has changed location since the last time sighted, the UserAgent may have to perform various actions. For example, if there are any clients of the UserAgent that have requested change-of-location callback notifications, then the UserAgent will need to generate the requested callbacks at this time. Another action that the UserAgent may perform when the user changes location is notify the Location Service of the user's changed location and request callbacks for changes occurring at the user's new location. (This only needs to be done when the user's policy specifies that the UserAgent may register with the Location Service so that others can find the user by proximity.) If the user has left the region serviced by a particular Locations Service, the UserAgent deregisters the user from that Location Service altogether, and registers the user with a Location Service covering the new region. The UserAgent may further attempt to contact terminal or other device agents at the new location, for example, to attempt to deliver an urgent message that was not deliverable at the previous location.

b) Callback from Location Service indicating that some other person or object has come near the user's current location. The UserAgent will update its list of persons and objects known to be currently near the user. The UserAgent may also need to perform various actions. For example, if the user's calendar specifies that a reminder message be given to the user when near a specified person, then the UserAgent may need to try to deliver an urgent (i.e., reminder) message to the user if the specified person is now near the user.

Callbacks from various services may also require the UserAgent to attempt to gain further information as part of acting on the callbacks. For example, if another user has registered with the Location Service in only a partial fashion (i.e., given a UserAgent RPC address to the Location Service, but not included an actual name or any other identifying information), then the UserAgent will try to request more information from the other user's UserAgent. (Theimer, 12:5-60.)

However, these excerpts simply describe the process by which a callback request can be made to a service that monitors state changes in which the user is interested. For example, as noted above, the Location Service may register a callback request, and then perform the callback when the location of a new object is updated. Likewise in Figure 11, the Active Badge Service registers a callback request (after step 310) and performs the callback (step 290) when the location of an identified badge is updated (step 286). In Figure 12, the Input Monitor Service registers a callback request (step 328) and performs the callback (step 320) when the location of an authenticated user is updated (step 316). Presumably, the UserAgent could register a callback with any of the three services listed above, depending on which state changes the user has an interest.

However, Theimer does not teach or even suggest monitoring requests from a client-source in order to detect possible circular references. While Theimer's disclosure regarding the use of callbacks and Applicants' claimed subject matter are very different, Applicants will attempt to discuss Applicants' claimed subject matter in the context of Theimer's disclosure. For example, a UserAgent may register a callback request with the Location Service for notification of any new objects moving into a particular office the user is in. Theimer does not teach or suggest, "during generating of the requested value of the first state attribute by the client-source" (in this example, during generating by the Location Service of the updated location of a new object moving into the particular office the user is in), "monitoring requests from the client-source" (here, monitoring callback requests *from* Theimer's Location Service) "for values of one or more indicated state attributes needed for the generating of the requested value of the first state attribute" (here, no state attributes are indicated whose values are needed for Theimer's updated location of a new object moving into the particular office the user is in).

Furthermore, with respect to the language in claims 11 and 25 that "when it is determined that a state attribute indicated in one of the monitored requests is the first state attribute, indicating a presence of a circular reference during the generating of the requested value of the first state attribute," Theimer similarly fails to include any teaching or suggestion of this claimed element. First, as noted above, while Theimer makes a callback ***to the Location Service***, Theimer does not monitor callback requests made ***by the Location Service***. However, even if

Theimer did monitor requests from the Location Service, the cited excerpts would still fail to teach Applicants' claimed subject matter. These excerpts simply describe the process by which a callback can be performed pursuant to an earlier request. To continue the previous example, the Location Service can perform the requested callback to UserAgent for notification of any new objects moving into a particular office the user is in. In response to the callback from the Location Service, UserAgent updates its internal state variables to reflect the updated location information for the new object and may seek further information in response to the callback. This sequence of callback request and callback performance is unrelated to monitoring for the presence of an endless loop of circular references while generating a particular attribute value, and thus is quite different from Applicants' claimed "indicating a presence of a circular reference during the generating of the requested value of the first state attribute."

In the prior Office Action, the Examiner responded to Applicants' prior explanation of Theimer's failings as follows:

> Using the registrations in the Location Service, circular references (i.e., callbacks) are determined anytime there is a detected change in the UserAgent state (Theimer: 12:5-12:60).

Thus, the Examiner appears to equate a simple notification via a callback mechanism with an endless loop of circular references involving multiple independent modules requesting information back-and-forth from each other. However, Theimer's callbacks are quite different from the recited "circular references". For instance, Applicants' circular references result from: "a request from a first client for a value of a first of the state attributes" (e.g., a request for a value of attribute 1); "determining a client-source able to generate and supply the requested value of the first state attribute" (e.g., CS 1 is determined able to generate and supply the value of attribute 1) "by using a value of at least one other state attribute" (e.g., CS 1 uses the value of attribute 2 in order to generate the value of attribute 1); "requesting the client-source to supply the requested value of the first state attribute" (e.g., requesting CS 1 to supply the value of attribute 1); "during generating of the requested value of the first state attribute by the client-source" (e.g., during generation of the value of attribute 1 by CS 1), "monitoring requests from

the client-source for values of one or more indicated state attributes needed for the generating of the requested value of the first state attribute" (e.g., requests from CS 1 for the value of attribute 2 needed to generate the value of attribute 1); "monitoring other requests for values of indicated state attributes needed for generating values of state attributes that are indicated in previously monitored requests" (e.g., requests from CS 2 for the value of attribute 1 needed for generating the value of attribute 2); and "determin[ing] that a state attribute indicated in one of the monitored requests is the first state attribute" (e.g., the request from CS 2 for the value of attribute 1 needed to generate the value of attribute 2 is the attribute 1 first requested from CS 1 in the first claim element).

Thus, Applicants strongly object to the Examiner's equating of Theimer's callbacks with Applicants' circular references, as Theimer's callbacks appear to be unrelated to the recited "circular references" of Applicants' claims. If the Examiner insists on maintaining the rejections, the Examiner is respectfully requested to point out exactly where in Theimer's disclosure there is support for the Examiner's position that Theimer's callbacks are in any way related to Applicants' indication of a circular reference.

Although the language of independent claims 27, 42, 56, and 57 are not identical to that of claims 11 or 25, each of these claims recites language similar to that of one or both of claims 11 and 25. For example, the independent claims recite the following:

claim 27 recites "indicating presence of a circular reference when it is determined that a state attribute indicated in one of the monitored requests is the first state attribute";

claim 42 recites "determining that a circular reference exists when it is determined that a module is to generate another value of the first context attribute such that the generating of the another value is caused by the generating of the first value of the first context attribute";

claim 56 recites "determining that a circular reference exists when it is determined that a module is to generate another value of the first context attribute such that the generating of the another value is caused by the generating of the first value of the first context attribute"; and

claim 57 recites "determining that a circular reference exists when it is determined that a module is to generate another value of the first context attribute such that the generating of the another value is caused by the generating of the first value of the first context attribute."

Since Theimer does not teach, suggest or motivate these claim elements, claims 27-29 and 42-57 are each similarly patentable over Theimer.

Accordingly, since Theimer lacks any teaching, suggestion or motivation to monitor for and detect circular references during generation of state attribute values, all of the pending claims are patentable over Theimer for at least this reason. Furthermore, the pending dependent claims recite various additional features that further render those claims patentable over the cited references, but that are not discussed here for the sake of brevity.

Conclusion

In light of the above remarks, Applicants respectfully submit that all of the pending claims are allowable. Applicants therefore respectfully request the Examiner to reconsider this application and timely allow all pending claims. If the Examiner has any questions or believes a telephone conference would expedite prosecution of this application, the Examiner is encouraged to call the undersigned at (206) 694-4860.

The Director is authorized to charge any additional fees due by way of this Amendment, or credit any overpayment, to our Deposit Account No. 19-1090.

Respectfully submitted,
SEED Intellectual Property Law Group PLLC

/James A. D. White/
James A. D. White
Registration No. 43,985

JDW:jaa

701 Fifth Avenue, Suite 6300
Seattle, Washington 98104-7092
Phone: (206) 622-4900
Fax: (206) 682-6031